



CERTIFIED COPY OF
PRIORITY DOCUMENT

P 9800-US

대한민국 특허청

KOREAN INTELLECTUAL
PROPERTY OFFICE

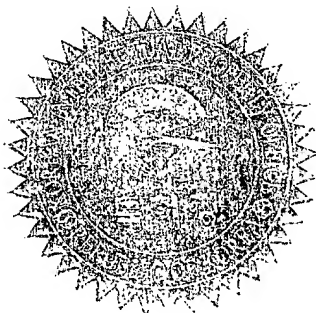
별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto
is a true copy from the records of the Korean Intellectual
Property Office.

출원번호 : 특허출원 2000년 제 33107 호
Application Number

출원년월일 : 2000년 06월 12일
Date of Application

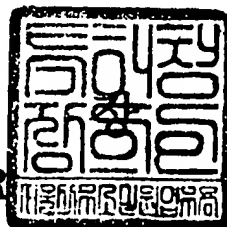
출원인 : 삼성전자 주식회사
Applicant(s)

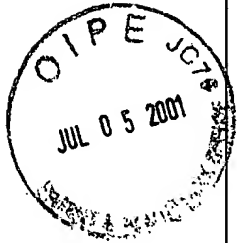


2001 년 06 월 14 일

특 허 청

COMMISSIONER





040
6-25

PATENT

Atty. Docket No. 678-693 (P9800)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

#2/10

APPLICANT(S): Jae-Yoel KIM et al.

SERIAL NO.: 09/879,688

GROUP: Art Unit - Not yet assigned

FILED: June 12, 2001

FOR: APPARATUS AND METHOD FOR
ENCODING AND DECODING TFCI
IN A MOBILE COMMUNICATION SYSTEM

Dated: July 2, 2001

Assistant Commissioner for Patents
Washington, D.C. 20231

TRANSMITTAL OF PRIORITY DOCUMENT

Sir:

Attached is a certified copy of Korean Appln. No. 33107/2000 filed on
June 12, 2000 from which priority is claimed under 35 U.S.C. §119.

Respectfully submitted,

Paul J. Farrell
Reg. No. 33,494
Attorney for Applicant(s)

DILWORTH & BARRESE, LLP
333 Earle Ovington Blvd.
Uniondale, NY 11553
(516) 228-8484

CERTIFICATE OF MAILING UNDER 37 C.F.R. §1.8(a)

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail, postpaid in an envelope addressed to the: Assistant Commissioner for Patents, Washington, D.C. 20231 on July 2, 2001.

Dated: June 2, 2001

Paul J. Farrell

BEST AVAILABLE COPY

【서류명】	특허출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	0001
【제출일자】	2000.06.12
【국제특허분류】	H04M
【발명의 명칭】	무선통신시스템의 채널 부호화/복호화 장치 및 방법
【발명의 영문명칭】	apparatus and method for channel coding and decoding i wireless communication system
【출원인】	
【명칭】	삼성전자 주식회사
【출원인코드】	1-1998-104271-3
【대리인】	
【성명】	이건주
【대리인코드】	9-1998-000339-8
【포괄위임등록번호】	1999-006038-0
【발명자】	
【성명의 국문표기】	김재열
【성명의 영문표기】	KIM, Jae Yoel
【주민등록번호】	700219-1047637
【우편번호】	435-042
【주소】	경기도 군포시 산본2동 백두아파트 960동 1401호
【국적】	KR
【발명자】	
【성명의 국문표기】	이현우
【성명의 영문표기】	LEE, Hyun Woo
【주민등록번호】	630220-1709811
【우편번호】	441-390
【주소】	경기도 수원시 권선구 권선동 택산 아파트 806동 901호
【국적】	KR
【발명자】	
【성명의 국문표기】	박성일
【성명의 영문표기】	PARK, Seong I I I
【주민등록번호】	680519-1481421

【우편번호】	435-040
【주소】	경기도 군포시 산본동 설악아파트 859동 2206호
【국적】	KR
【발명자】	
【성명의 국문표기】	최호규
【성명의 영문표기】	CHOI, Ho Kyu
【주민등록번호】	681204-1787524
【우편번호】	137-030
【주소】	서울특별시 서초구 잠원동 56-2 신반포27차 351-603
【국적】	KR
【취지】	특허법 제42조의 규정에 의하여 위와 같이 출원합니다. 대 리인 주 (인) 이권
【수수료】	
【기본출원료】	20 면 29,000 원
【가산출원료】	27 면 27,000 원
【우선권주장료】	0 건 0 원
【심사청구료】	0 항 0 원
【합계】	56,000 원
【첨부서류】	1. 요약서·명세서(도면)_1통

【요약서】**【요약】**

시간 분할 듀플렉스 방식을 사용하는 부호분할 다중접속 통신시스템의 전송 포맷 조합 표시 비트를 전송하기 위한 TFCI비트 코딩 장치 및 방법에 관한 것으로 최대 10개의 TFCI비트를 입력하여 48개의 코딩된 TFCI심볼을 생성하는 장치 및 방법을 제시한다. 상기 코딩된 TFCI심볼은 두 개의 서브프레임으로 구성되는 하나의 10ms 프레임에 4개의 부분으로 나뉘어져서 전송된다.

【대표도】

도 8

【색인어】

TFCI, TDD, coding, decoding walsh, mask

【명세서】**【발명의 명칭】**

무선통신시스템의 채널 부호화/복호화 장치 및 방법{apparatus and method for channel coding and decoding in wireless communication system}

【도면의 간단한 설명】

도1은 종래의 협대역 시간분할 듀플렉스 방식을 사용하는 부호분할 다중접속 통신 시스템의 프레임 구조를 설명하는 도면

도2는 종래의 전송 포맷 조합 표시비트를 전송하기 위한 송신기 구조를 도시하는 도면.

도 3은 종래의 수신기의 간략한 구성을 도시하는 도면.

도 4에는 10비트로 표현되는 확장된 TFCI의 오류정정부호화 방식의 예를 도시하는 도면.

도 5은 선형 오류정정부호의 부호화 구조를 도시하는 도면.

도 6은 마스크함수의 생성방법을 설명하는 도면.

도 7a 및 도 7b는 본 발명에 따른 부호기를 도시하는 도면.

도 8은 상기 도 7a의 부호기의 동작 따른 플로우 차트를 도시하는 도면.

도 9는 상기 도 7의 부호기에 대응하는 복호기를 도시하는 도면.

도 10은 상기 도 9에서 비교기940의 동작을 플로우 차트로 도시하는 도면.

도 11은 본 발명에 따른, 길이 64인 월시부호들과 상기 월시부호에 모두 1을 취한 64개의 부호들, 총 128개의 부호에 구해진 7개의 길이 64인 마스크함수들을 조합한 수열

들을 부어호로 사용하는데, 이에 대한 구조를 도시하는 도면.

도 12는 상기 도 7b의 부호기의 동작에 대한 플로우 차트를 도시하는 도면.

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

- <13> 본 발명은 부호분할다중접속시스템의 TFCI부호발생기에 관한 장치 및 방법에 관한 것으로, 특히 TFCI부호발생기의 장치 및 방법에 관한 것이다.
- <14> 일반적으로 부호분할을 수행하는 이동통신 시스템(이하 IMT2000 시스템이라 칭한다)에서는 하나의 물리적 채널안에 음성서비스, 화상서비스, 데이터서비스와 같은 여러 가지 서비스의 프레임이 같이 전송한다. 상기 서비스들의 프레임은 고정된 데이터 전송율로 전송되거나 가변적인 전송율로 전송된다. 고정된 전송율로 전송되는 서로 다른 서비스들은 각각의 서비스의 프레임의 확산레이트(spreading rate)를 수신측에 일일이 알려줄 필요가 없지만 가변적인 전송율로 전송되는 서비스들은 데이터의 전송율이 서비스 중간에 달라질 수 있으므로 각각의 서비스의 프레임의 확산레이트를 수신측에 알려줘야 한다. 일반적으로 IMT2000 시스템에서는 데이터의 전송속도와 데이터의 확산레이트가 반비례한다. 상기 각각의 서비스가 사용하는 프레임의 전송속도가 다른 경우에 현재 전송되고 있는 서비스의 조합을 알려주는 역할 하는 것이 TFCI이고, TFCI는 각각의 서비스가 올바르게 수신될 수 있도록 한다. NB-TDD(Narrowband TDD)의 경우 상기 TFCI의 사용 예는 도1과 같다. 특히, NB-TDD(Narrowband TDD)에서 고속전송을 위하여 8PSK(Phase shift

Keying) modulation을 사용하고, 이 때, 상기 TFCI의 값은 길이 48인 부호로 부호화되어 전송된다.

<15> 도 2는 프레임 전송하는 송신기의 구조를 도시한다. 도 2를 참조하여 송신구조를 설명하면 먼저 TFCI비트가 TFCI부호기 200에 입력되면, 부호화 되어 멀티플렉서210에 입력된다. 이때, 도1의 하나의 슬롯 내에 포함된 data심볼, SS심볼 및 TPC심볼로 구성되는 기타 신호들이 멀티플렉서210에 입력되어 상기 TFCI부호 심볼들과 도1의 미드엠블 신호를 제외한 구조와 같이 멀티플렉스 되어진 후 출력되면 상기 멀티플렉스 되어진 신호들은 채널확산기 220에 입력되어 직교부호로 채널확산 되어진다. 상기 직교부호로 확산된 신호는 스크램블러230에 입력되어 스크램블링 코드와 스크램블링 되어진 후 멀티플렉서 240에 입력된다. 이때, Midamble도 상기 멀티플렉서 240에 입력되어 상기 스크램블링 되어진 신호와 도1의 구조와 같은 출력을 같도록 멀티플렉스 되어진 후 출력된다. 상기 멀티플렉스 210 및 240을 컨트롤러의 제어를 받아 상기 도1과 같은 프레임 구조를 가지도록 제어를 받으며 상기 컨트롤러는 도면에서 생략하였다.

<16> 도 3은 종래의 수신기의 간략한 구성을 도시하고 있다. 도 3를 참조하여 수신구조를 설명하면 먼저 수신된 신호는 디멀티플렉서 340에 입력되어 디 멀티플렉싱 되어진 후 Midamble을 분리하고, 그 외의 디 멀티플렉싱 되어진 신호들은 디스크램블러 330에 입력되어 상기 스크램블링 코드로 디스크램블링 되어진다. 그러면, 상기 디스크램블링 되어진 신호들은 다시 채널 역확산기 320에 입력되어 상기 직교부호로 역확산 되어진 후 디 멀티플렉서 310에 입력되어 TFCI심볼과 그외의 신호를 분리하고, 상기 분리되어진 TFCI 심볼들은 TFCI 복호기300에 입력되어 복호화 되어진 후 TFCI 정보비트로 출력되어진다.

<17> 상기 TFCI비트의 값은 서비스들의 조합에 따라 1-2비트로 표현되어 1-4가지의 조합을

나타내거나, 3-5비트로 표현되어 8-32개의 조합을 나타내거나, 6-10비트로 표현되어 64-1024개의 조합을 표현한다. 상기 TFCI의 값은 수신단에서 각 서비스들의 프레임을 해석하기 위해 반드시 필요한 정보이므로, 전송 오류가 발생한다면 수신단에서 각 서비스들의 프레임을 올바르게 해석하지 못하는 경우가 발생한다. 따라서 상기 TFCI의 값은 전송중에 오류가 발생한다할지라도 수신단에서 TFCI의 전송중에 발생하는 오류를 정정할 수 있도록 오류정정부호를 사용하여 부호화된다.

<18> 도 4에는 10비트로 표현되는 확장된 TFCI의 오류정정부호화 방식의 예가 도시되어 있다.

<19> 상기 도4에 도시된 바와 같이 1-1024 사이의 TFCI값은 10비트로 표현되어, (32,10) 확장된 리드물러 (Extended Reed Muller) 부호기 400으로 입력되어, 32심볼의 TFCI 부호어로 출력된다. 그러면 상기 32심볼의 출력된 부호심볼들은 반복기410에 입력되고, 반복기 410은 입력된 부호심볼들의 짝수번째 심볼들은 그대로 출력하고 홀수번째 심볼들만을 반복하여 총 48개의 부호심볼들을 출력한다. 도4에서 입력정보비트는 TFCI값의 10비트 표현형식으로 10비트가 안되는 TFCI값은 MSB(Most significant bit, 제일 왼쪽의 bit)부터 0의 값으로 채워져 6비트 길이로 만들어진다. 상기 도4의 (32,10) 확장된 리드물러 (Extended Reed Muller) 부호기는 대한민국 출원 특허번호 1999-27932에 게재되어 있다.

<20> 상기 도4의 (32,10) 확장된 리드물러 (Extended Reed Muller) 부호기의 부호간 최단거리는 12이다. 또한, 상기 도4의 반복기를 거치고 나면 (48,10)부호로써 최단거리가 16이 된다. 일반적으로 이진 선형 부호(Binary Linear Codes)의 오류 정정 능력은 이진 선형 부호의 각 부호간 최단 거리에 따라 결정되는데 최적부호(optimal code)가 되기 위한 이진 선형 부호의 입력과 출력값에 따른 부호간의 최단 거리는 [1]과 같다.

- <21> **참조문헌[1] An Updated Table of Minimum-Distance Bounds for Binary Linear Codes
- <22> (A.E. Brouwer and Tom Verhoeff, IEEE Transactions on information Theory, VOL 39, NO. 2, MARCH 1993)**
- <23> 상기 도 4에서 전송되는 TFCI의 값이 10비트이고, 부호화되는 값이 48비트임을 생각하면 참조문헌 1에서 요구하는 최적부호의 각 부호간의 최단거리는 19~20이다. 그러나 상기 부호기의 부호간 최단거리는 16이 되므로, 상기 도3의 오류정정방식부호화는 최적부호를 가지지 못한다. 상기 도5의 오류정정방식부호화에서 최적부호를 가지지 못한다면 동일한 채널환경에서 TFCI의 오류확률이 커진다. 따라서, TFCI의 오류가 발생하여 데이터프레임의 전송률을 잘못 판단하고 데이터프레임을 복호화하면 데이터프레임의 오류율을 증가시키게 된다. 따라서, TFCI를 부호화하는 오류정정부호기는 오류율을 최소화하는 것이 중요하다.

【발명이 이루고자 하는 기술적 과제】

- <24> 따라서 본 발명의 목적은 (48,10) 코딩 장치 및 그 디코딩 장치를 제공함에 있다.
- <25> 본 발명의 다른 목적은 시간분할 듀플렉스 방식의 협대역 부호분할 다중접속 시스템의 전송 포맷 조합 표시 비트를 코딩하는 장치 및 방법을 제공한다.
- <26> 본 발명의 또 다른 목적은 시간분할 듀플렉스 방식의 협대역 부호분할 다중접속 시스템의 전송 포맷 조합 표시 비트를 디코딩 하는 장치 및 방법을 제공한다.
- <27> 상기 목적들을 달성하기 위한, 부호분할다중접속시스템의 부호화기는, 심볼 값이 1인 심볼을 지속적으로 출력하는 1비트 발생기와, 길이 64의 월시부호들을 심볼 단위로 발생하

는 기저 월시부호발생기와, 길이 64의 제1-제3 기저 마스크 함수들을 심볼단위로 발생하는 마스크 생성기와, 상기 1비트 생성기와 상기 기저 월시부호 발생기 및 상기 마스크 생성기로부터 발생하는 심볼들과 상기 심볼들에 대응하는 TFCI 정보비트들을 각각 승산하여 출력하는 승산기들과, 상기 승산기들의 출력을 가산하여 마스크된 TFCI 정보에 해당하는 부호심볼들을 출력하는 가산기로 구성함을 특징으로 한다.

【발명의 구성 및 작용】

- <28> 본 발명은 상기한 바와 같이 TFCI비트를 사용하는 CDMA 시스템에서 TFCI를 부호화하는 방법에 있어서 최적부호를 생성할 수 있는 방법으로 확장된 리드물러(Extended Reed Muller)를 CDMA시스템에 적용하는 하는 것이다.
- <29> 선형 오류정정부호(Linear Error Correcting Code)의 성능을 나타내는 척도(measure)로서는 오류정정부호의 부호어(codeword)의 해밍 거리(Hamming distance)의 분포가 있는데, 이는 각각의 부호어에서 0이 아닌 심볼의 개수를 의미한다. 즉, 0111이 어떤 부호어라면 이 부호어에 포함된 1의 개수, 즉, 해밍거리는 3이다. 이 때, 여러 부호어의 해밍거리 값들 중 가장 작은 값을 최소거리 d_{min} (minimum distance)라고 칭한다. 선형 오류정정부호(Linear Error Correcting Code)에 있어서 상기의 최소거리가 클수록 오류정정성능이 우수하다.
- <30> 상기의 확장된 리드물러(Extended Reed Muller)는 특정 시퀀스(sequence)와 m -시퀀스의 합으로 이루어지는 시퀀스로부터 유추해 낼 수 있는데, 상기의 시퀀스들의 합을 원소로하는 시퀀스군을 선형 오류정정부호로 사용하기 위해서는 시퀀스군의 최소거리가

커야한다. 이러한 시퀀스군으로는 카자미 시퀀스(Kasami sequence)군, 골드 시퀀스(Gold sequence)군이나, 커독코드(Kerdock Code)군과 같은 시퀀스군들이 있다. 상기의 시퀀스들은 전체길이 $L=2^{2m}$ 일 때, 최소거리가 $(2^{2m} - 2^m)/2$ 이고, $L=2^{2m+1}$ 일 때, 최소거리가 $2^{2m} - 2^m$ 이다. 즉, 전체길이가 64일 때, 최소거리는 28이다.

<31> 본 발명에서는 상기와 같은 시퀀스군을 사용하여 우수한 성능을 가지는 선형 오류정정부호인 확장된 오류정정부호를 생성하는 생성방법 및 장치에 대해 설명하기로 한다.

<32> 코드이론(Coding Theory)에서, 상기 m -시퀀스를 순환 쉬프트 하여 구한 m -시퀀스군을 월시부호로 만드는 열치환 함수는 존재한다. 이 때, 상기와 같은 특정 시퀀스(sequence)와 m -시퀀스의 합으로 이루어지는 시퀀스들을 상기의 m -시퀀스군을 월시부호로 만드는 열치환함수로 열치환 하게 되면, m -시퀀스군은 월시부호가 되고, 특정 시퀀스(sequence)들은 상기의 월시부호와의 합의 최소거리가 상기의 성질을 만족하게 되는데, 이를 이하 마스크함수라 칭하기로 한다. 도 5은 상기와 같은 선형 오류정정부호의 부호화 구조를 나타낸다.

<33> 하기에서는 카자미 시퀀스군을 사용하여, $(2^n, n+k)$ 코드(단, $k=1, \dots, n+1$)를 생성하는 경우 상기와 같은 마스크함수의 생성방법을 설명한다. 실제로, 카자미 시퀀스는 서로 다른 특정의 m -시퀀스의 합으로 표현되어진 다는 것은 널리 알려진 사실이다. 따라서, 상기의 $(2^n, n+k)$ 코드를 생성하기 위해서는, 먼저, 길이 2^n-1 인 카자미 시퀀스를 생성하여야 하는데, 생성다항식 $f_1(x)$ 으로 생성되어지는 m -시퀀스와 상기 m -시퀀스를 $2^{(n/2)+1}$ 단위로 데시메이션(Decimation)한 길이 $2^{(n/2)}-1$ 인 수열을 $2^{(n/2)+1}$ 번 반복한 시퀀스의 합은 카자미시퀀스가 된다. 또한, 각각의 m -시퀀스 $m(t)$ 는 생성다항식이 정해지면

<수학식1>과 같이 트레이스 함수(Trace function)를 이용하여 구할 수 있다.

<34> 【수학식 1】

$$m_1(t) = \text{Tr}(A \alpha^t), t=0,1,\dots,30$$

<35>

$$\text{단, } \text{Tr}(a) = \sum_{k=0}^{n-1} a^{2^k}, a \in GF(2^n) \text{ 이다.}$$

<36> 상기 <수학식1>에서 A는 m-시퀀스의 초기치에 따라 결정되는 값이다.

<37> 도6은 상기의 시퀀스군중 카자미 시퀀스군을 사용하여, $(2^n, n+k)$ 코드(즉, $n+k$ 비트의 정보비트가 입력되면 2^n 비트의 부호화심볼이 출력되는)를 생성하는 경우 상기과 같은 마스크함수의 생성과정을 나타낸다. 카자미 시퀀스는 서로다른 특정의 m-시퀀스의 합으로 표현되어진 다는 것은 널리 알려진 사실이다. 따라서, 상기의 $(2^n, n+k)$ 코드를 생성하기 위해서는, 먼저, 길이 $2^{n/2}-1$ 인 카자미 시퀀스를 생성하여야 하는데, 생성다항식 $f_1(x)$ 으로 생성되어지는 m-시퀀스와 상기 m-시퀀스를 $2^{(n/2)+1}$ 단위로 데시메이션(Decimation)한 길이 $2^{(n/2)-1}$ 인 수열을 $2^{(n/2)+1}$ 번 반복한 시퀀스의 합은 카자미시퀀스가 된다.

<38> 도6을 살펴보면, 610 단계에서는 상기의 <수학식1>에 의해 생성다항식 $f_1(x)$ 로 생성되어지는 m-시퀀스 $m_1(t)$ 와 상기 m-시퀀스를 $2^{(n/2)+1}$ 단위로 데시메이션(Decimation)한 길이 $2^{(n/2)-1}$ 인 수열을 $2^{(n/2)+1}$ 번 반복한 시퀀스 $m_2(t)$ 를 구한다. 그러면, 620단계에서는 상기의 상기의 m-시퀀스 $m_1(t)$ 를 하기의 <수학식 2>에서 나타난 월시부호로 만드는 열치환 함수 $\sigma(t)$ 를 구한다.

<39> 【수학식 2】

$$\sigma : \{0,1,2,\dots, 2^n - 2\} \rightarrow \{1,2,\dots, 2^n - 1\}$$

$$\langle 40 \rangle \quad \sigma(t) = \sum_{i=0}^{n-1} m_i(t) 2^{n-1-i}$$

$\langle 41 \rangle$ 그러면, 630단계에서는 상기 m-시퀀스 $m_2(t)$ 를 0부터 30번까지 순환 쉬프트(cyclic shift)시켜 얻을 수 있는 7개의 시퀀스군을 상기의 $m_1(t)$ 를 월시부호로 만드는 열치환 함수 $\sigma(t)$ 의 역함수를 이용한 $\sigma^{-1}(t) + 2$ 로 열치환한 후, 각각의 시퀀스 맨 앞부분에 0을 덧붙임으로써 길이 2^n 로 만들어서 길이 2^n 인 2^n-1 개의 시퀀스군 $d_i(t)$, $i=0, \dots, 2^n-1$, $t=1, \dots, 2^n$,을 생성한다. 상기과 같이 730단계에서 생성되어지는 시퀀스군은 <수학식 3>과 같이 수식으로 표현할 수 있다.

$\langle 42 \rangle$ 【수학식 3】

$$\{d_i(t) \mid t=1, \dots, 2^n, \quad i=0, \dots, 2^{\frac{n}{2}}-2\}$$

$$\langle 43 \rangle \quad d_i(t) = \begin{pmatrix} 0, & \text{if } t=1 \\ m_{\sigma(t+i-2)}, & \text{if } t=1, 2, 3, \dots, 2^n \end{pmatrix}$$

$\langle 44 \rangle$ 상기에서 구해진 $d_i(t)$ 들은 상기 마스크 함수들로 7개의 마스크로 사용할 수 있다.

$\langle 45 \rangle$ 상기에서 구해진 $d_i(t)$ 들의 성질 중 한가지 성질은 상기의 마스크들중 두 개의 서로 다른 마스크들을 더하면 $2^{(n/2)}-1$ 개마스크중 다른 하나의 마스크가 된다. 더 일반화시켜서 전부 0인 마스크를 포함하여 상기의 $2^{(n/2)}-1$ 개의 마스크들은 $2^{(n/2)}-1$ 개의 마스크중 특정한 n 개의 마스크의 임의의 합으로 모두 표현되어질 수 있다.

$\langle 46 \rangle$ 상기 $(2^n, n+k)$ 코드를 생성할 때, 총 필요로하는 부호어(Code word)의 개수는 모든

경우의 입력 정보비트의 가지수인 2^{n+k} 개이다. 이 때, 길이 2^n 인 상호직교(Biorthogonal) 부호어의 개수는 $2^n \times 2 = 2^{n+1}$ 이고, 이 때, $(2^n, n+k)$ 코드를 생성하기 위해 필요한 마스크의 개수는 $(2^{n+k}/2^{n+1}) - 1 = 2^{k-1} - 1$ 개이다. 또한, 이 때, $2^{k-1} - 1$ 개의 마스크는 상기와 유사한 성질에 의해 $k-1$ 개의 마스크의 임의의 합으로 모두 표현되어질 수 있다. 따라서, 상기의 $k-1$ 개의 마스크를 고르는 방법이 필요하다. 상기의 $k-1$ 개의 마스크를 선택하는 방법을 설명하면, 상기 도 7의 730단계에서 $m_2(t)$ 를 0부터 $2^{(n/2)-1}$ 번까지 순회(cyclic shift)시켜 시퀀스군을 생성하는데, 이 때, $m_2(t)$ 를 i 번 순회(cyclic shift)시킨 m -시퀀스는 상기 <수학식1>을 사용하여 표현하면 $Tr(\alpha^i \cdot \alpha')$ 가 된다. 즉, $m_2(t)$ 를 0부터 6번까지 순회(cyclic shift)시켜 시퀀스군은 초기치 A 가 $1, \alpha, \dots, \alpha^{2^n-2}$ 에 따라서 생성되어지는 시퀀스들이다. 이 때, 갈로아체의 원소 $1, \alpha, \dots, \alpha^{2^n-2}$ 중, 선형독립인 $k-1$ 개의 원소를 찾아서 상기 $k-1$ 개의 원소를 초기치로 하는 시퀀스들을 선택하여 상기 도7의 결과값들을 찾아 선택하면 상기와 같이 $k-1$ 개의 마스크의 임의의 합으로 $2^{k-1} - 1$ 개 마스크 모두를 생성할 수 있다. 상기의 과정중 선형독립의 조건을 수학식으로 나타내면 하기의 <수학식4>과 같다

<47> 【수학식 4】

$\alpha_1, \dots, \alpha_{k-1} : \text{선형 독립}$

<48>
$$\Leftrightarrow c_1\alpha_1 + c_2\alpha_2 + \dots + c_{k-1}\alpha_{k-1} \neq 0, \quad \forall c_1, c_2, \dots, c_{k-1}$$

<49> 상기의 일반화된 마스크 함수생성방법을 구체적인 예를 들어 설명하기 위해서 하기에서는 카자미 시퀀스군을 사용하여, $(64, 10)$ 코드를 생성하는 경우 상기와 같은 마스크

함수의 생성방법을 상기의 일반화된 과정에 따라 도 6을 참조하여 설명한다. 실제로, 카자미 시퀀스는 서로다른 특정의 m-시퀀스의 합으로 표현되어진 다는 것은 널리 알려진 사실이다. 따라서, 상기의 (64,10)코드를 생성하기 위해서는, 먼저, 길이 63인 카자미 시퀀스를 생성하여야 하는데, 생성다항식 $x^6 + x + 1$ 과 상기 m-시퀀스를 $2^{(n/2)+1}$ 단위로 데시메이션(Decimation)한 길이 $2^{(n/2)}-1$ 인 수열을 $2^{(n/2)+1}$ 번 반복한 시퀀스의 합은 카자미시퀀스가 된다. 또한, 각각의 m-시퀀스 $m(t)$ 는 생성다항식이 정해지면 <수학식1>과 같이 트레이스 함수(Trace function)를 이용하여 구할 수 있다.

<50> 【수학식 5】

$$m_1(t) = \text{Tr}(A \alpha^t), t=0,1,\dots,63$$

<51> 단, $\text{Tr}(\alpha) = \sum_{n=0}^4 \alpha^{2^n}$, $\alpha \in GF(2^5)$ 이다.

<52> 상기 <수학식1>에서 A는 m-시퀀스의 초기치에 따라 결정되는 값이다.

<53> 도 6은 상기의 시퀀스군중 카자미 시퀀스군을 사용하여, (64,10)코드(즉, 10비트의 정보비트가 입력되면 64비트의 부호화심볼이 출력되는)를 생성하는 경우 상기와 같은 마스크함수의 생성과정을 나타낸다. 카자미 시퀀스는 서로다른 특정의 m-시퀀스의 합으로 표현되어진 다는 것은 널리 알려진 사실이다. 따라서, 상기의 (64,10)코드를 생성하기 위해서는, 먼저, 길이 63인 카자미 시퀀스를 생성하여야 하는데, 6차 생성다항식 $x^6 + x + 1$ 과 상기 m-시퀀스를 $2^{(n/2)+1}$ 단위로 데시메이션(Decimation)한 길이 $2^{(n/2)}-1$ 인 수열을 $2^{(n/2)+1}$ 번 반복한 시퀀스의 합은 카자미시퀀스가 된다.

<54> 도 6을 살펴보면, 610 단계에서는 상기의 <수학식1>에 의해 생성다항식 $x^6 + x + 1$ 로 생성되어지는 m-시퀀스 $m_1(t)$ 와 상기 m-시퀀스 $m_1(t)$ 를 $2^{(n/2)+1}$ 단위로 데시메이션

(Decimation)한 길이 $2^{(n/2)-1}$ 인 수열을 $2^{(n/2)+1}$ 번 반복한 시퀀스 $m_2(t)$ 를 구한다. 그러면, 620단계에서는 상기의 상기의 m -시퀀스 $m_1(t)$ 를 하기의 <수학식 2>에서 나타난 월시부호로 만드는 열치환 함수 $\sigma(t)$ 를 구한다.

<55> 【수학식 6】

$$\sigma : \{0, 1, 2, \dots, 63\} \rightarrow \{1, 2, \dots, 64\}$$

<56>

$$\sigma(t) = \sum_{i=0}^4 m_i(t) 2^{4-i}$$

<57>

그러면, 630단계에서는 상기 m -시퀀스 $m_2(t)$ 를 0부터 6번까지 순환 쉬프트(cyclic shift)시켜 얻을 수 있는 7개의 시퀀스군을 상기의 $m_1(t)$ 를 월시부호로 만드는 열치환 함수 $\sigma(t)$ 의 역함수를 이용한 $\sigma^{-1}(t) + 2$ 로 열치환한 후, 각각의 시퀀스 맨 앞부분에 0을 덧붙임으로써 길이 64로 만들어서 길이 64인 7개의 시퀀스군 $d_i(t)$, $i=0, \dots, 6$, $t=1, \dots, 32$ 을 생성한다. 상기과 같이 630단계에서 생성되어지는 시퀀스군은 <수학식 3>과 같이 수식으로 표현할 수 있다.

<58> 【수학식 7】

$$\{d_i(t) | t=1, \dots, 64, i=0, \dots, 6\}$$

<59>

$$d_i(t) = \begin{cases} 0, & \text{if } t=1 \\ m_d(t+i-2), & \text{if } t=2, 3, \dots, 64 \end{cases}$$

<60>

상기에서 구해진 $d_i(t)$ 들은 상기 마스크 함수들로 7개의 마스크로 사용할 수 있다.

<61>

상기에서 구해진 $d_i(t)$ 들의 성질 중 한가지 성질은 상기의 마스크들중 두 개의 서로 다른 마스크들을 더하면 7개마스크 중 다른 하나의 마스크가 된다. 더 일반화시켜서

상기의 7개의 마스크들은 7개의 마스크중 특정한 5개의 마스크의 임의의 합으로 모두 표현되어질 수 있다.

<62> 상기 (64,10)코드를 생성할 때, 총 필요로하는 부호어(Code word)의 개수는 모든 경우의 입력 정보비트의 가지수인 $2^{10} = 1024$ 개이다. 이 때, 길이 64인 상호직교(Biorthogonal) 부호어의 개수는 $64 \times 2 = 128$ 이고, 이 때, (64,10)코드를 생성하기 위해 필요한 마스크의 개수는 $(1024/128) - 1 = 7$ 개이다. 또한, 이 때, 7개의 마스크는 상기와 유사한 성질에 의해 3개의 마스크의 임의의 합으로 모두 표현되어질 수 있다. 따라서, 상기의 3개의 마스크를 고르는 방법이 필요하다. 상기의 3개의 마스크를 선택하는 방법을 설명하면, 상기 도6의 630단계에서 $m_2(t)$ 를 0부터 6번까지 순회(cyclic shift)시켜 시퀀스군을 생성하는데, 이 때, $m_2(t)$ 를 i 번 순회(cyclic shift)시킨 m -시퀀스는 상기 <수학식1>을 사용하여 표현하면 $Tr(\alpha' \cdot \alpha')$ 가 된다. 즉, $m_1(t)$ 를 0부터 6번까지 순회(cyclic shift)시켜 시퀀스군은 초기치 A 가 $1, \alpha, \dots, \alpha^6$ 에 따라서 생성되어지는 시퀀스들이다. 이 때, 갈로아체의 원소 $1, \alpha, \dots, \alpha^6$ 중, 선형독립인 3개의 원소를 찾아서 상기 3개의 원소를 초기치로 하는 시퀀스들을 선택하여 상기 도7의 결과값들을 찾아 선택하면 상기와 같이 3개의 마스크의 임의의 합으로 7개 마스크 모두를 생성할 수 있다. 상기의 과정중 선형독립의 조건을 수학식으로 나타내면 하기의 <수학식8>과 같다.

<63> 【수학식 8】

$\alpha, \beta, \gamma, \delta$: 선형 독립

<64> $\Leftrightarrow c_1\alpha + c_2\beta + c_3\gamma + c_4\delta \neq 0, \quad \forall c_1, c_2, c_3, c_4$

<65> 실제로 갈로아체 $GF(2^3)$ 에서 1,

α , α^2 은 상기와 같은 4개의 선형독립인 원소로 널리 알려진 다항식 기저(polynomial basis)이다. 따라서, 상기 다항식 기저에 따라서 구해진 3개의 마스크 함수M1,M2,M4는 다음과 같다.

<66> M1 = 0011010101101111101000110000011011110110010100111001111111000101

<67> M2 = 0100011111010001111011010111101101111011000100101101000110111000

<68> M4 = 0001100011100111110101001101010010111101101111010111000110001110

<69> 하기에 나타날 실시예에서는 상기에서 구해진 마스크를 이용한 부호기 및 복호기에 대한 실시예를 나타낸다.

<70> 제1실시예

<71> 상기 제 1실시예에서는 상기의 생성방법에 따른 부호화 장치 및 방법을 제공한다.
도 7은 상기 제1실시예에 따른 부호기를 나타낸 도면이다.

<72> 도 7을 참조하면 10비트의 입력 정보비트들이 입력되면, 각각의 비트

a0,a1,a2,a3,a4,a5, a6,a7,a8,a9들은 승산기 740,741,742,743,744,745,746,747,748,749에 입력된다. 이 때, 월시부호 발생기 710에서는 길이 64인 월시부호

W1,W2,W4,W8,W16,w32이 동시에 출력되는데, 월시부호

W1=01인 심볼들이 승산기 740에 출력되고, 한 심볼이 승산기 840으로 입력될 때마다, 승산기 740에 입력된 a0와 승산되어져 가산기 760에 입력되고, 월시부호

W2=0011001100110011001100110011001100110011001100110011001100110011001100110011인 심볼들이 승산기 741에 출력되고, 한 심볼이 승산기 741으로 입력될 때마다, 승산기 741에 입력된 a1와 승산되어져 가산기 760에 입력되고, 월시부호

W4=0000111100001111000011110000111100001111 1000011110000111100001111인 심볼들이 승산기 742에 출력되고, 한 심볼이 승산기 742으로 입력될 때마다, 승산기 742에 입력된 a2와 승산 되어져 가산기 760에 입력되고, 월시부호 W8=000000000

111111110000000001111111110000000001111111100000000011111111인 심볼들이 승산기 743에 출력되고, 한 심볼이 승산기 743으로 입력될 때마다, 승산기 743에 입력된 a3와 승산되어져 가산기 760에 입력되고, 월시부호 W16=

0000000000000000011111111111111100000000000000001111111111111111인 심볼들이 승산기 744에 출력되고, 한 심볼이 승산기 744으로 입력될 때마다, 승산기 744에 입력된 a4와 승산되어져 가산기 760에 입력되고, 월시부호

W32=0000000000000000000000000000000001111111111111111 1111111111111111인 심볼들이 승산기 745에 출력되고, 한 심볼이 승산기 745로 입력될 때마다, 승산기 745에 입력된 a5와 승산되어져 가산기 760에 입력된다.

<73> 또, 1비트 발생기 700에서 항상 1인 심볼들이 승산기 746에 출력되고, 한 심볼이 승산기 746으로 입력될 때마다, 승산기 740에 입력된 a6와 승산되어져 가산기 760에 입력된다. 이것은 상호 직교부호를 발생하기 위한 것이다.

<74> 마스크 생성기 720에서는 길이 64인 마스크 함수 M1,M2,M4가 동시에 출력되는데, 마스크 함수 M1=00110101011011110100011000001101111011001010011100111111000101인 심볼들이 승산기 747에 출력되고, 한 심볼이 승산기 747으로 입력될 때마다, 승산기 747

에 입력된 a7와 승산되어져 가산기 760에 입력되고, 마스크 함수

M2=010001111101000111101101011110110111101100010010110100011 0111000인 심볼들이 승산기 748에 출력되고, 한 심볼이 승산기 748으로 입력될 때마다, 승산기 748에 입력된 a8와 승산되어져 가산기 760에 입력되고, 마스크 함수 M4=0001100011100111110101001101010010111101101111010111000110001110인 심볼들이 승산기 749에 출력되고, 한 심볼이 승산기 749으로 입력될 때마다, 승산기 749에 입력된 a9와 승산되어져 가산기 760에 입력된다.

<75> 그러면, 가산기 760은 승산기 740,741,742,743,744,745,746,747,748,749으로부터 출력된 심볼들을 모두 가산하여 출력한다. 이때, 출력되어진 64개의 부호화 심볼들은 다시 심볼 천공기 770으로 입력되어 16심볼이 천공되어진 후 48심볼만이 출력되어진다. 상기 (48,10)부호기는 상기 (64,10) 부호기로부터 16심볼을 천공함으로써 얻어질 수 있다. 이 때, 상기 16개의 천공심볼 위치에 따라서, 상기 부호기의 최소거리가 달라지는데, 하기에 나타나는 16개의 천공위치는 모든 16개의 위치에 대한 조합 중에서도 우수한 성능을 나타내는 위치를 나타내고, 하기에 나타나는 천공위치를 사용하면 최소거리가 18을 가지게 되고, 웨이트 분포도 우수하다.

<76> {0, 4, 8, 13, 16, 20, 27, 31, 34, 38, 41, 44, 50, 54, 57, 61}

<77> {0, 4, 8, 13, 16, 21, 25, 28, 32, 37, 43, 44, 49, 52, 56, 62}

<78> {0, 4, 8, 13, 16, 21, 25, 31, 32, 37, 43, 44, 49, 52, 56, 61}

<79> {0, 4, 8, 13, 18, 21, 25, 30, 35, 36, 40, 46, 50, 53, 57, 62}

<80> {0, 4, 8, 13, 18, 21, 25, 30, 35, 37, 40, 47, 50, 53, 57, 62}

<81> {0, 4, 8, 13, 19, 22, 27, 30, 33, 36, 41, 44, 49, 55, 58, 61}

<82> {0, 4, 8, 13, 19, 22, 27, 30, 33, 36, 41, 44, 50, 52, 56, 63}

<83> {0, 4, 8, 13, 19, 22, 27, 30, 33, 36, 41, 44, 50, 52, 58, 61}

<84> {0, 4, 8, 13, 16, 20, 27, 31, 34, 38, 41, 44, 50, 54, 57, 61}

<85> 상기 도 7의 승산기 740, 741, 742, 743, 744, 745, 746, 747, 748, 749는 입력 TFCI비트에 의하여 해당하는 직교부호 기저들 및 마스크 기저들 및 상기 직교부호를 상호직교부호로 만들기 위한 모두 1인 값의 출력을 제어하는 것이다.

<86> 도 8은 상기 도 7과 같은 결과를 얻기 위한 플로우 차트를 나타낸다. 도 8을 참조 하면, 먼저 800단계에서 10비트의 정보비트열 a_0, a_1, \dots, a_9 를 입력하고, 부호기에서 최종적으로 출력할 부호심볼을 나타낼 64개의 변수인 $code[i]$ 를 0으로 초기화 하고, 변수 i 를 0으로 초기화한다.

<87> 그러면, 810단계에서는 첫 번째 정보비트 a_0 이 1일 때, 일시부호 $W1 = 01$ 를 길이 64인 부호심볼열 변수 $code[i]$ 에 배타적 가산해주는 과정이고, 이과정이 끝나면, 812단계에서는 두 번째 정보비트 a_1 가 1일 때, 일시부호 $W2 = 0011001100110011001100110011001100110011001100110011001100110011$ 를 길이 64인 부호심볼열 변수 $code[i]$ 에 배타적 가산해주는 과정이고, 이 과정이 끝나면, 814단계에서는 세 번째 정보비트 a_2 가 1일 때, 일시부호

W4=00001111000011110000111100001111000011110000111100001111를 길이 64인 부호심볼열 변수 code[]에 배타적 가산해주는 과정이고, 이 과정이 끝나면, 816단계에서는 네번째 정보비트 a4가 1일 때, 월시부호

W8=0000000011111111000000001111111100000000111111110000000011111111를 길이 64인 부호심볼열 변수 code[]에 배타적 가산해주는 과정이고, 이 과정이 끝나면, 818단계에서는 다섯번째 정보비트 a4가 1일 때, 월시부호 W16=0000000000000000

11111111111111110000000000000000111111111111111를 길이 64인 부호심볼열 변수 code[]에 배타적 가산해주는 과정을 진행한다. 그러면, 820단계에서는 여섯번째 정보비트 a5가 1일 때, 월시부호

W32=000000000000000000000000000000001111111111111111111111111111를 길이 64인 부호심볼열 변수 code[]에 배타적 가산해주는 과정을 진행하고, 822단계를 진행하여 일곱번째 정보비트 a6이 1인지를 판단하고, 1이면 변수 j를 0으로 초기화하고, j번째 변수인 code[j]와 1을 배타적 가산한다. 그리고, j가 보호심볼의 마지막인지를 판단하기 위하여 j가 63인지를 판단하고, 아니면 j를 1만큼 증가시켜, 상기의 과정을 반복한다. 상기의 과정은 정보비트 a6이 1일 때, 전부 1인 길이 64인 수열을 길이 64인 부호심볼열에 배타적 가산해주는 과정이다. 따라서, 상기의 과정이 64번 진행되면, 변수 j는 63이 되어 j가 63인지를 판단하는 과정에서 824단계로 진행하게 된다. 824단계에서는 여덟번째 정보비트 a8가 1일 때, 마스크 함수 M1 =

001101010110111110100011000001101111011001010011100111111000101를 길이 64인 부호심볼열 변수 code[]에 배타적 가산해주는 과정을 진행하고, 826단계에서는 아홉

번째 정보비트 a_8 이 1일 때, 마스크 함수 $M_2 =$

0100011111010001111011010111101101111011000100101101000 110111000를 길이 64인 부호
심볼열 변수 $code[]$ 에 배타적 가산해주는 과정을 진행하고, 826단계에서는 열번째 정보비
트 a_9 이 1일 때, 마스크 함수 $M_4 = 000110001110011111010100110101001011110110$

1111010111000110001110를 길이 64인 부호심볼열 변수 $code[]$ 에 배타적 가산해주는 과정
을 진행한다. 상기의 과정이 끝나면 상기의 10비트의 정보비트에 대응되는 10개의 길이
64인 수열 $W_1, W_2, W_4, W_8, W_{16}, W_{32}, 1, M_1, M_2, M_4$ 들중 1인 정보비트들에 대응되는 수
열들만을 모드 배타적가산한 부호화 심볼 변수의 값 $code[]$ 이 출력된다.

<88> 상기 (64,10)부호기는 길이 64인 64개의 월시부호와 모든 월시부호의 심볼에 전부
1을 더한 실수의 경우 -1을 곱한), 총 128개의 부호와 3개의 마스크함수의 조합에 구해
지는 총 7가지의 마스크함수의 조합으로 나타나며, 따라서, 총 부호어의 수는
1024개이다. 또한, 상기 1024개의 부호어들 중 길이 64인 64개의 월시부호와 모든 월시
부호의 심볼에 전부 1을 더한 실수의 경우 -1을 곱한), 총 128개의 부호와 상기 3개의
마스크함수중 2개의 마스크함수의 조합에 구해지는 총 4가지의 마스크함수의 조합으로 나
타나는 (64,9)부호기, 상기 1024개의 부호어들 중 길이 64인 64개의 월시부호와 모든 월
시부호의 심볼에 전부 1을 더한 실수의 경우 -1을 곱한), 총 128개의 부호와 상기 3개의
마스크함수중 임의의 1개의 마스크의 조합에 구해지는 총 2가지의 마스크함수의 조합으
로 나타나는 (64,8)부호기들은 모두 최소거리 28을 가진다. 상기와 같은 (64,9)부호기는
상기 도 7a의 마스크함수 생성기에

서 출력되는 3가지 마스크함수중 한 부분의 입력과 출력을 중단시킴으로써 구현되어질 수 있고, (64,8)부호기는 상기 도7a의 마스크함수 생성기에서 출력되는 3가지 마스크함수중 두 부분의 입력과 출력을 중단시킴으로써 구현되어질 수 있다. 상기와 같은 작용을 함으로써 상기의 부호기는 입력정보비트수에 따라 유동적으로 부호화가 가능하며, 부호기의 성능을 좌우하는 최소거리를 최대한 높임으로써 우수한 성능을 갖게 된다.

<89> 상기의 부호기는 길이 64인 월시부호들과 상기 월시부호에 모두 1을 취한 64개의 부호들, 총 128개의 부호에 상기와 같은 방법으로 구해진 7개의 길이 64인 마스크함수들을 조합한 수열들을 부어호로 사용하는데, 이에 대한 구조는 도 11에서 나타난다.

<90> 도 9는 부호기에 대응하는 복호기를 도시하는 도면이다. 상기 도 9을 살펴보면, 먼저 길이 48인 +1/-1값을 가지는 TFCI심볼에 해당하는 수신신호에 상기 부호기에서 적용한 천공위치에 0을 삽입하여 입력신호의 길이를 64로 만들어진 수신신호 $r(t)$ 는 7개의 승산기 901,902,...,907와 상관도 계산기920에 입력된다. 그러면, 마스크 생성기910은 상기 3개의 마스크 기저에 의해서 생성될 수 있는 모든 마스크 7개를 생성하여 승산기 901,902,...,907에 출력한다. 이 때, 승산기 901은 입력된 수신신호 $r(t)$ 와 마스크 생성기 910으로부터 출력된 마스크함수 M_1 을 승산하여 상관도 계산기 921에 출력하고, 승산기 902는 입력된 수신신호 $r(t)$ 와 마스크 생성기 910으로부터 출력된 마스크함수 M_2 를 승산하여 상관도 계산기942에 출력하고, 이런 식으로 하여 승산기 907은 입력된 수신신호 $r(t)$ 와 마스크 생성기 910으로부터 출력

된 마스크함수 M7를 승산하여 상관도 계산기927에 출력하면 수신신호 자신과 가능한 7개의 모든 마스크함수들이 수신신호 $r(t)$ 와 승산되어진, 총 8가지의 신호들이 8개의 상관도 계산기에 입력된다. 그러면, 상관도 계산기는 입력된 수신신호 $r(t)$ 를 64개의 모든 길이 64인 월시부호와 모든 월시부호의 심볼에 전부 -1(이진수의 경우 1)을 곱한 부호들, 총 128가지와의 128가지 상관도를 구하여 상관도의 값이 가장 큰값과, 그 때 계산되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스인 0을 상관도 비교기940에 출력한다. 이 때, 상기 상관도 계산기의 인덱스는 상기 상관도 계산기에 입력된 신호가 수신신호에 몇번째 마스크 함수가 승산되어져 입력되었는지를 나타내는 마스크 함수의 인덱스와 동일하다. 그러나, 마스크 인덱스가 0이라함은 아무런 마스크도 곱해지지 않았음을 의미한다. 그리고, 이와 동시에 상관도 계산기1은 입력된 수신신호 $r(t)$ 와 마스크 함수 M1을 승산한 신호를 64개의 모든 길이 64인 월시부호와 모든 월시부호의 심볼에 전부 -1(이진수의 경우 1)을 곱한 부호들, 총 128가지와의 128가지 상관도를 구하여 상관도의 값이 가장 큰값과, 그 때 계산되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스인 1을 상관도 비교기940에 출력하고, 상관도 계산기2는 입력된 수신신호 $r(t)$ 와 마스크 함수 M2를 승산한 신호를 64개의 모든 길이 64인 월시부호와 모든 월시부호의 심볼에 전부 -1(이진수의 경우 1)을 곱한 부호들, 총 128가지와의 128가지 상관도를 구하여 상관도의 값이 가장 큰값과, 그 때 계산되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스인 2를 상관도 비교기940에 출력하고, 이런식으로하여, 상관도 계산기15는 입력된 수신신호 $r(t)$ 와 마스크 함수 M7를 승산한 신호를 64개의 모든 길이

64인 월시부호와 모든 월시부호의 심볼에 전부 -1(이진수의 경우 1)을 곱한 부호들, 총 128가지와의 128가지 상관도를 구하여 상관도의 값이 가장 큰값과, 그 때 계산되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스인 7을 상관도 비교기940에 출력한다.

<91> 그러면, 상관도비교기는 상기 상관도계산기에서 입력된 8가지의 최대 상관값을 비교하여 가장 최대인 상관값을 구하면, 그 상관값에 대해 해당 상관도 계산기로부터 입력되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스, 즉, 수신신호 $r(t)$ 와 송신되어진 마스크함수의 인덱스를 가지고, 수신신호에 대한 복호신호를 계산하여 출력한다.

<92> 도 10은 상기의 비교기940의 동작을 플로우 차트로 나타낸 것이다. 도10을 참조하여 설명하면 먼저 1000단계에서는 회수를 나타내는 인덱스 I은 1로, 검색하려는 최대값, 월시부호 인덱스, 마스크인덱스들은 0으로 초기화 한다. 그러면, 1010단계에서는 1번째 상관도 계산기에서 입력된 920에서 출력된 상관도의 1번째 최대값, 월시부호 인덱스, 마스크 인덱스들을 입력한다. 그러면, 1020에서 상기 1번째 최대값과 최대값을 비교하여 1번째 최대값이 크면 1030단계로 진행하여 최대값에 1번째 최대값을, 월시부호 인덱스에는 1번째 상관도 계산기로부터 입력된 월시부호 인덱스를, 마스크 인덱스에는 1번째 상관도 계산기로부터 입력된 마스크 인덱스를 차례로 저장하고, 1040단계를 진행하고, 1020에서 상기 1번째 최대값과 최대값을 비교하여 1번째 최대값이 작으면 1040단계를 바로 진행한다. 이 때, 1040단계에서는 회수를 나타내는 인덱스 i가 상기 상관도계산기의 개수인 8인지를 판단한

다. 1040단계에서는 회수를 나타내는 인덱스 i 가 상기 상관도계산기의 개수인 8이 아니면 1060단계를 진행하여 회수를 나타내는 인덱스 i 를 1씩 증가시키고, 다시 1010단계를 진행하여 2번째 상관도 계산기에서 입력된 922에서 출력된 상관도의 2번째 최대값, 월시부호 인덱스, 마스크 인덱스들을 입력하고 상기와 같은 과정을 반복한다. 상기와 같이 반복하다가 8번째 상관도 계산기로부터 입력된 상관값이 모두 비교되면 이 때의 회수를 나타내는 i 가 8일 것이므로 1040단계에서 i 가 8인지를 판단하여 통과하면 1050단계로 진행하여 상기 변수인 월시부호 인덱스와 마스크 인덱스에 대응하는 복호비트들을 출력한다.

<93> 상기의 제1실시예는 (48,10) 부호화기를 도실했다. 하기의 제 2실시예에서는 상기 도 7a와 달리 월시부호생성기, 1생성기 및 마스크 생성기에서 미리 상기 천공패턴을 적용하여 16심볼을 천공한 후 48심볼을 출력하는 부호화기를 도식한다.

<94> 제2실시예

<95> 상기 제 2실시예에서는 상기의 생성방법에 따른 부호화 장치 및 방법을 제공한다. 상기의 부호화 장치의 구조는 제1실시예의 부호화구조와 유사하다. 단, 1비트 발생기, 월시부호 발생기와 마스크 생성기에서 출력되는 수열들은 상기 제1 실시예에서의 수열들의 0,4,8,13,16,20,27,31,34,38, 41,44,50,54,57,61번째의 항을 천공함으로써 길이 48을 가진다. 따라서, 도 7b를 참조하여 제2실시예의 부호화기를 설명하면 10비트의 입력 정보비트들이 입력되면, 각각의 비트 $a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9$ 들은 승산기 740, 741, 742, 743, 744, 745, 746, 747, 748, 749에 입력된다. 이 때, 월시부호 발생기 710에서는 길이 48인 천공된

월시부호 W1, W2, W4, W8, W16이 동시에 출력되는데, 천공된 월시부호

W1=101101101001101101010010011011001101011011001001인 심볼들이 승산기 740에 출력되고, 한 심볼이 승산기 740으로 입력될 때마다, 승산기 740에 입력된 a0와 승산되어져 가산기 760에 입력되고, 천공된 월시부호

W2=011011011011011011001001001001011011001001011011인 심볼들이 승산기 741에 출력되고, 한 심볼이 승산기 741로 입력될 때마다, 승산기 741에 입력된 a1와 승산되어져 가산기 760에 입력되고, 천공된 월시부호

W4=000111000111000111000111000111000111000111000111인 심볼들이 승산기 742에 출력되고, 한 심볼이 승산기 742로 입력될 때마다, 승산기 742에 입력된 a2와 승산되어져 가산기 760에 입력되고, 천공된 월시부호

W8=000000111111000000111111000000111111000000111111인 심볼들이 승산기 743에 출력되고, 한 심볼이 승산기 743으로 입력될 때마다, 승산기 743에 입력된 a7와 승산되어져 가산기 760에 입력되고, 천공된 월시부호 W16=000000000000111111111111000000000000

111111111111인 심볼들이 승산기 744에 출력되고, 한 심볼이 승산기 744로 입력될 때마다, 승산기 744에 입력된 a4와 승산되어져 가산기 760에 입력되고, 천공된 월시부호

W32=00000000000000000000 000011111111111111111111111111인 심볼들이 승산기 745에 출력되고, 한 심볼이 승산기 745로 입력될 때마다, 승산기 745에 입력된 a5와 승산되어져 가산기 760에 입력된다.

<96> 또, 1비트 발생기 700에서 항상 1인 심볼들이 승산기 740에 출력되고, 한 심볼이 승산기 746으로 입력될 때마다, 승산기 746에 입력된 a6와 승산되어져 가산기 760에 입력된다.

<97> 마스크 생성기 820에서는 길이 48인 천공된 마스크 함수 M1,M2,M4이 동시에 출력되는데, 천공된 마스크 함수 M1=0111 0111 0111 0100 1100 0011 1110 1000 1011 1011 1110 0001인 심볼들이 승산기 847에 출력되고, 한 심볼이 승산기 847로 입력될 때마다, 승산기 847에 입력된 a7와 승산되어져 가산기 860에 입력되고, 천공된 마스크 함수 M2=1001 1110 1001 1101 0101 1101 0111 0100 1010 1110 0111 1100인 심볼들이 승산기 848에 출력되고, 한 심볼이 승산기 848로 입력될 때마다, 승산기 848에 입력된 a8와 승산되어져 가산기 860에 입력되고, 천공된 마스크 함수 M4=0010 0011 0011 1011 0011 0010 1011 1111 1101 0110 0110 0110인 심볼들이 승산기 849에 출력되고, 한 심볼이 승산기 849로 입력될 때마다, 승산기 849에 입력된 a9와 승산되어져 가산기 860에 입력된다.

<98> 그러면, 가산기 860은 승산기 840,841,842,843,844,845,846,847,848,849으로부터 출력된 심볼들을 모두 가산하여 출력한다.

<99> 도 12는 상기 부호기의 동작에 대한 플로우 차트를 나타낸다. 도12을 참조하면, 먼저 1200단계에서 10비트의 정보비트열 a0,a1,...,a9를 입력하고, 부호기에서 최종적으로 출력할 부호심볼을 나타낼 48개의 변수인 code[]를 0으로 초기화 하고, 변수 i를 0으로 초기화한다. 그러면, 1210단계에서는 첫 번째 정보비트 a0이 1일 때, 일시부호 W1 = 10110110100110110101001001101100110101101100 1001를 길이 48인 부호심볼열 변수 code[]에 배타적 가산해주는 과정이고, 이 과정이 끝나면, 1212단계에서는 두 번째 정보비트 a1가 1일 때, 일시부호 W2 = 01101101101101101100100100100101101100100101 1011를 길이 48인 부호심볼열 변수 code[]에 배타적 가산해주는 과정이고, 이 과정이 끝나면, 1214단계에서는 세 번째 정보비트 a2가 1일 때, 일시부호 W4 = 000111000111000111000111000111000111000111000111를 길이 48인 부호심볼열 변수

code[]에 배타적 가산해주는 과정이고, 이 과정이 끝나면, 1216단계에서는 네번째 정보 비트a3이 1일 때, 월시부호 W8=

<100> 000000111111000000111111000000111111000000111111를 길이 48인 부호심볼열 변수

code[]에 배타적 가산해주는 과정이고, 이 과정이 끝나면, 1218단계에서는 다섯번째 정보비트a4가 1일 때, 월시부호 W16=000000000000011111111111000000000000111111111111를

길이 48인 부호심볼열 변수 code[]에 배타적 가산해주는 과정을 진행한다. 그러면, 1220단계에서는 여섯번째 정보비트a5가 1일 때, 월시부호

M1=0000000000000000000000001111111111111111111111111를 길이 48인 부호심볼열 변수

code[]에 배타적 가산해주는 과정을 진행하고, 1222단계를 진행하여 일곱번째 정보비트 a6이 1인지를 판단하고, 1이면 변수 j를 0으로 초기화하고, j번째 변수인 code[j]와 1을 배타적 가산한다. 그리고, j가 보호심볼의 마지막인지를 판단하기 위하여 j가 47인지를 판단하고, 아니면 j를 1만큼 증가시켜, 상기의 과정을 반복한다. 상기의 과정은 정보비트 a6이 1일 때, 전부 1인 길이 48인 수열을 길이 48인 부호심볼열에 배타적 가산해주는 과정이다. 따라서, 상기의 과정이 48번 진행된다면,

변수 j 는 47이 되어 j 가 47인지를 판단하는 과정에서 1224단계로 진행하게 된다.

그러면, 1224단계에서는 여덟번째 정보비트 a_7 가 1일 때, 마스크 함수 $M_1 = 0111\ 0111$

$0111\ 0100\ 1100\ 0011\ 1110\ 1000\ 1011\ 1011\ 1110\ 0001$ 를 길이 48인 부호심볼열 변수

$code[]$ 에 배타적 가산해주는 과정을 진행하고, 1226단계에서는 아홉번째 정보비트 a_8 이 1

일 때, 마스크 함수 $M_2 = 1001\ 1110\ 1001\ 1101\ 0101\ 1101\ 0111\ 0100\ 1010\ 1110\ 0111$

1100 를 길이 48인 부호심볼열 변수 $code[]$ 에 배타적 가산해주는 과정을 진행하고, 1228단

계에서는 열번째 정보비트 a_9 이 1일 때, 마스크 함수 $M_4 = 0010\ 0011\ 0011\ 1011\ 0011$

$0010\ 1011\ 1111\ 1101\ 0110\ 0110\ 0110$ 를 길이 48인 부호심볼열 변수 $code[]$ 에 배타적 가

산해주는 과정을 진행한다. 상기의 과정이 끝나면 상기의 10비트의 정보비트에 대응되는

10개의 길이 32인 수열 $W_1, W_2, W_4, W_8, W_{16}, W_{32}, 1, M_1, M_2, M_4$ 들중 1인 정보비트들
에 대응되는 수열들만을 모드 배타적가산한 부호화 심볼 변수의 값 $code[]$ 이 출력된다.

<101> 상기 (48,10)부호기는 상기 제1실시예에 (48,10)부호기의 모든 부호어들에 대해서
0,4,8,13,16,20,27,31,34,38, 41,44,50,54,57,61번째 심볼들을 천공하여 만든 1024개의
부호어로 구성되어있다. 따라서, 총 부호어의 수는 1024개이다. 또한, 상기 1024개의 부
호어들 중 64개의 길이 64인 월시부호에서 0,4,8,13,16,20,27,31,34,38,
41,44,50,54,57,61번째 심볼들이 천공된 길이 48인 월시부호와 상기 천공된 모든 월시부
호의 심볼에 전부 1을 더한 실수의 경우 -1을 곱한), 총 128개의 부호와 상기 3개의 천
공된 마스크함수중 임의의 2개의 마스크의 조합에 구해지는 총4가지의 마스크함수의 조
합으로 나타나는 (48,9)부호기, 상기

1024개의 부호어들 중 길이 48인 64개의 천공된 월시부호와 모든 천공된 월시부호의 심볼에 전부 1을 더한 실수의 경우 -1을 곱한), 총 128개의 부호와 상기 3개의 천공된 마스크함수중 임의의 1개의 마스크의 조합에 구해지는 총2가지의 마스크함수의 조합으로 나타나는 (48,8)부호기들은 모두 최소거리 18을 가진다.

<102> 상기와 같은 (48,9)부호기는 상기 도7b의 마스크함수 생성기에서 출력되는 4가지 마스크함수중 한 부분의 입력과 출력을 중단시킴으로써 구현되어질 수 있고, (48,8)부호기는 상기 도7b의 마스크함수 생성기에서 출력되는 4가지 마스크함수중 두 부분의 입력과 출력을 중단시킴으로써 구현되어질 수 있고, 또한, (48,7)부호기는 상기 도7b의 마스크함수 생성기에서 출력되는 3가지 마스크함수중 세 부분의 입력과 출력을 중단시킴으로써 구현되어질 수 있다. 상기와 같은 작용을 함으로써 상기의 부호기는 입력정보비트수에 따라 유동적으로 부호화가 가능하며, 부호기의 성능을 좌우하는 최소거리를 최대한 높임으로써 우수한 성능을 갖게 된다.

<103> 도 9은 상기 부호기에 따른 복호기를 도시한다. 도 9을 살펴보면, 먼저 길이 48인 +1/-1값을 가지는 TFCI심볼에 해당하는 수신신호에 상기 부호기에서 적용한 천공위치에 0을 삽입하여 입력신호의 길이를 64로 만들어진 수신신호 $r(t)$ 는 7개의 승산기 901,902,...,907와 상관도 계산기 920에 입력된다. 그러면, 마스크 생성기 910은 7개의 모든 경우의 마스크를 생성하여 승산기 901,902,...,907에 출력한다. 이 때, 승산기 901은 입력된 수신신호 $r(t)$ 와 마스크 생성기 910으로부터 출력된 마스크함수 M1을 승산하여 상관도 계산기 921에 출력하고, 승산기 902는 입력된 수신신호 $r(t)$ 와 마스크 생성기 910으로부터 출력된 마스크함수 M2를 승산하여

상관도 계산기2 942에 출력하고, 이런식으로하여 승산기 907은 입력된 수신신호 $r(t)$ 와 마스크 생성기 910으로부터 출력된 마스크함수 $M7$ 을 승산하여 상관도 계산기7 927에 출력하면 수신신호 자신과 가능한 7개의 모든 마스크함수들이 수신신호 $r(t)$ 와 승산되어진, 총 8가지의 신호들이 8개의 상관도 계산기0-7에 입력된다. 그러면, 상관도 계산기0은 입력된 수신신호 $r(t)$ 를 64개의 모든 길이 48인 월시부호와 모든 월시부호의 심볼에 전부 -1(이진수의 경우 1)을 곱한 부호들, 총 128가지와의 128가지 상관도를 구하여 상관도의 값이 가장 큰값과, 그 때 계산되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스인 0을 상관도 비교기940에 출력한다. 이 때, 상기 상관도 계산기의 인덱스는 상기 상관도 계산기에 입력된 신호가 수신신호에 몇번째 마스크 함수가 승산되어져 입력되었는지를 나타내는 마스크 함수의 인덱스와 동일하다. 그러나, 마스크 인덱스가 0이라함은 아무런 마스크도 곱해지지 않았음을 의미한다. 그리고, 이와 동시에 상관도 계산기1은 입력된 수신신호 $r(t)$ 와 마스크 함수 $M1$ 을 승산한 신호를 64개의 모든 길이 48인 월시부호와 모든 월시부호의 심볼에 전부 -1(이진수의 경우 1)을 곱한 부호들, 총 128가지와의 128가지 상관도를 구하여 상관도의 값이 가장 큰값과, 그 때 계산되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스인 1을 상관도 비교기940에 출력하고, 상관도 계산기2는 입력된 수신신호 $r(t)$ 와 마스크 함수 $M2$ 를 승산한 신호를 64개의 모든 길이 48인 월시부호와 모든 월시부호의 심볼에 전부 -1(이진수의 경우 1)을 곱한 부호들, 총 128가지와의 128가지 상관도를 구하여 상관도의 값이 가장 큰값과, 그 때 계산되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스인 2를

상관도 비교기940에 출력하고, 이런식으로하여, 상관도 계산기7는 입력된 수신신호 $r(t)$ 와 마스크 함수 $M7$ 을 승산한 신호를 64개의 모든 길이 48인 월시부호와 모든 월시부호의 심볼에 전부 -1 (이진수의 경우 1)을 곱한 부호들, 총 128가지와의 128가지 상관도를 구하여 상관도의 값이 가장 큰값과, 그 때 계산되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스인 7을 상관도 비교기940에 출력한다.

<104> 그러면, 상관도비교기는 상기 상관도 비교기0-7에서 입력된 8가지의 최대 상관값을 비교하여 가장 최대인 상관값을 구하면, 그 상관값에 대해 해당 상관도 계산기로부터 입력되어진 월시부호의 인덱스, 그리고, 상관도 계산기의 인덱스, 즉, 수신신호 $r(t)$ 와 승산되어진 마스크함수의 인덱스를 가지고, 수신신호에 대한 복호신호를 계산하여 출력한다. 도 10은 상기의 비교기990의 동작을 플로우 차트로 나타낸 것이다. 도10을 참조하여 설명하면 먼저 1000단계에서는 회수를 나타내는 인덱스 I 은 1로, 검색하려는 최대값, 월시부호 인덱스, 마스크인덱스들은 0으로 초기화 한다. 그러면, 1010단계에서는 1번째 상관도 계산기에서 입력된 920에서 출력된 상관도의 1번째 최대값, 월시부호 인덱스, 마스크 인덱스들을 입력한다. 그러면, 1020에서 상기 1번째 최대값과 최대값을 비교하여 1번째 최대값이 크면 1030단계로 진행하여 최대값에 1번째 최대값을, 월시부호 인덱스에는 1번째 상관도 계산기로부터 입력된 월시부호 인덱스를, 마스크 인덱스에는 1번째 상관도 계산기로부터 입력된 마스크 인덱스를 차례로 저장하고, 1040단계를 진행하고, 1020에서 상기 1번째 최대값과 최대값을 비교하여 1번째 최대값이 작으면 1040단계를 바로 진행한다. 이

때, 1040단계에서는 회수를 나타내는 인덱스 i 가 상기 상관도계산기의 개수인16인지를 판단한다. 1040단계에서는 회수를 나타내는 인덱스 i 가 상기 상관도계산기의 개수인16이 아니면 1060단계를 진행하여 회수를 나타내는 인덱스 i 를 1씩 증가시키고, 다시 1010단계를 진행하여 2번째 상관도 계산기에서 입력된 922에서 출력된 상관도의 2번째 최대값, 월시부호 인덱스, 마스크 인덱스들을 입력하고 상기와 같은 과정을 반복한다. 상기과 같이 반복하다가 16번째 상관도 계산기로부터 입력된 상관값이 모두 비교되면 이 때의 회수를 나타내는 i 가 16일 것이므로 1040단계에서 i 가 16인지를 판단하여 통과하면 1050단계로 진행하여 상기 변수인 월시부호 인덱스와 마스크 인덱스에 대응하는 복호비트들을 출력한다.

【발명의 효과】

<105> 상술한 바와 같이 본 발명은 시간분할 듀플렉스 방식의 협대역 부호분할 다중접속 시스템에서 전송 포맷 조합 표시 비트를 효과적으로 코딩 및 디코딩할수 있다.

【특허청구범위】

【청구항 1】

부호분할다중접속시스템의 부호화기에 있어서,

심볼 값이 1인 심볼을 지속적으로 출력하는 1비트 발생기와,

길이 64의 월시부호들을 심볼 단위로 발생하는 기저 월시부호발생기와,

길이 64의 제1-제3 기저 마스크 함수들을 심볼단위로 발생하는 마스크 생성기와,

상기 1비트 생성기와 상기 기저 월시부호 발생기 및 상기 마스크 생성기로부터 발생하는 심볼들과 상기 심볼들에 대응하는 TFCI 정보비트들을 각각 승산하여 출력하는 승산기들과,

상기 승산기들의 출력을 가산하여 마스킹된 TFCI 정보에 해당하는 부호심볼들을 출력하는 가산기로 구성함을 특징으로 하는 부호분할다중접속시스템의 부호발생장치.

【청구항 2】

제1항에 있어서, 상기 마스크 생성기는,

제 1 기저 마스크 함수

$M1 = 001101010110111110100011000001101111011001010011100111111000101$ 와

제 2 기저 마스크 함수

$M2 = 0100011111010001111011010111101101111011000100101101000110111000$ 와

제 3기저 마스크 함수

$M4 = 0001100011100111110101001101010010111101101111010111000110001110$ 를 심볼 단위로 발생하는 것을 특징으로 하는 부호분할다중접속시스템의 부호발생장치.

【청구항 3】

부호분할다중접속시스템의 부호화 방법에 있어서,

심볼 값이 1인 심볼을 지속적으로 출력하는 1비트 발생과정과,

길이 64의 월시부호들을 심볼 단위로 발생하는 기저 월시부호발생과정과,

길이 64의 제1-제3 기저 마스크 함수들을 심볼단위로 발생하는 마스크 생성과정과,

상기 심볼값이 1인 심볼, 상기 길이 64인 월시부호에 해당하는 심볼, 상기 제1-제3 기저 마스크 함수들에 대응하는 심볼 및 상기 심볼들에 대응하는 TFCI 정보비트들을 각각 승산하여 출력하는 과정과,

상기 승산 결과들을 가산하여 마스크된 TFCI 정보에 해당하는 부호심볼들을 출력하는 가산과정을 포함하는 것을 특징으로 하는 방법.

【청구항 4】

제3항에 있어서,

제 1 기저 마스크 함수

$M1 = 001101010110111110100011000001101111011001010011100111111000101$ 이고,

제 2 기저 마스크 함수

M2 = 0100011111010001111011010111101101111011000100101101000110111000이며,

제 3기저 마스크 함수

M4 = 0001100011100111110101001101010010111101101111010111000110001110 인 것을 특징으로 하는 방법.

【청구항 5】

부호분할다중접속시스템의 복호화 장치에 있어서,
마스크 신호들을 생성하는 마스크 생성기와,
수신신호와 대응되는 상기 마스크 생성기로부터의 마스크 신호를 승산하는 승산기들과,
상기 수신신호 및 상기 수신신호와 상기 마스크 신호를 승산한 신호들을 상관하여 상관도가 가장 큰 값과 그때의 월시부호 인덱스를 출력하는 상관기들과,
상기 상관기들로부터의 최대 상관값들을 비교하여 가장 큰 상관값을 구하여 출력하는 상관비교기를 포함하는 것을 특징으로 하는 장치.

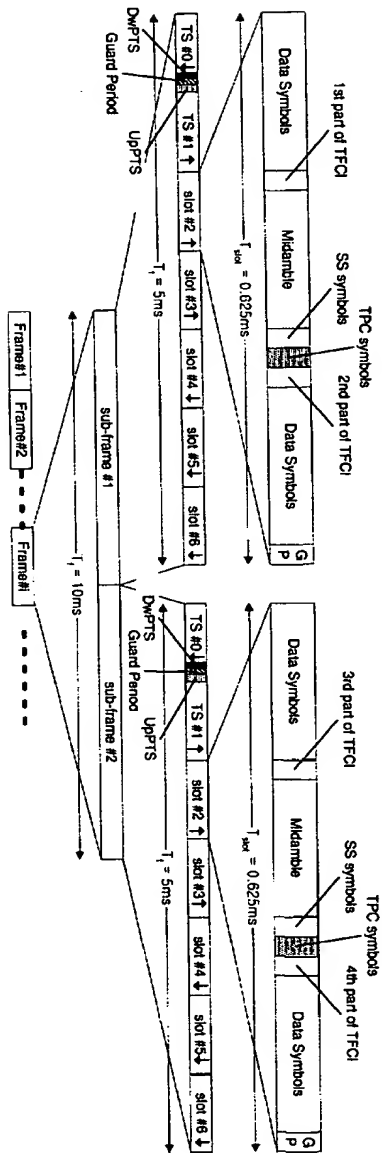
【청구항 6】

부호분할다중접속시스템의 복호화 방법에 있어서,
마스크 신호들을 생성하는 마스크 생성과정과,
수신신호와 대응되는 상기 마스크 신호를 승산하는 승산과정과,
상기 수신신호 및 상기 수신신호와 상기 마스크 신호를 승산한 신호들을 상관하여 상관도가 가장 큰 값과 그때의 월시부호 인덱스를 출력하는 상관과정과,

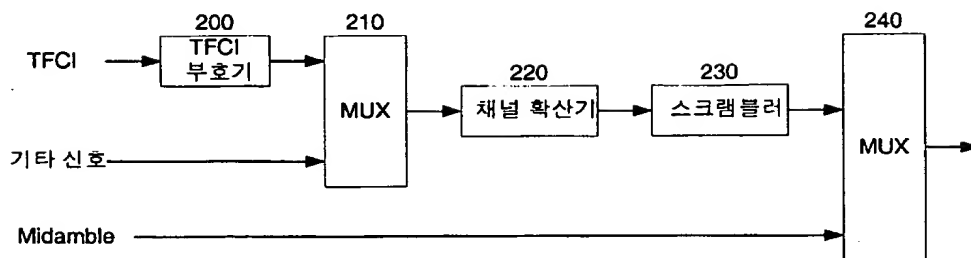
상기 최대 상관값들을 비교하여 가장 큰 상관값을 선택하는 상관비교 과정을 포함하는
것을 특징으로 하는 방법.

【도면】

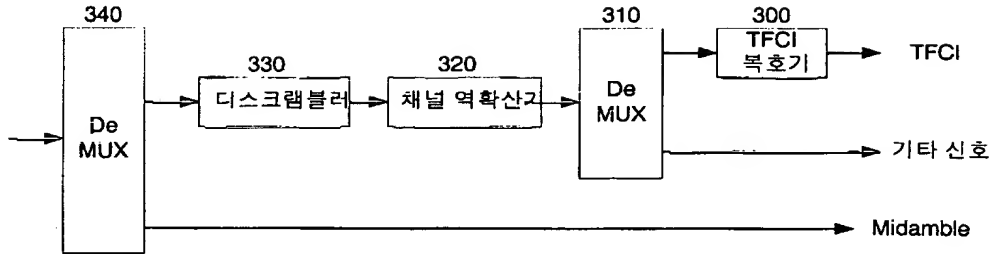
【도 1】



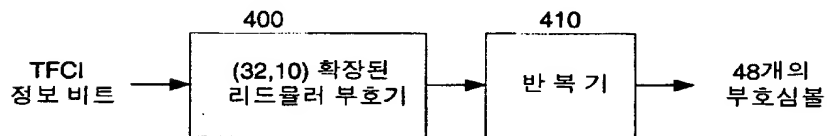
【도 2】



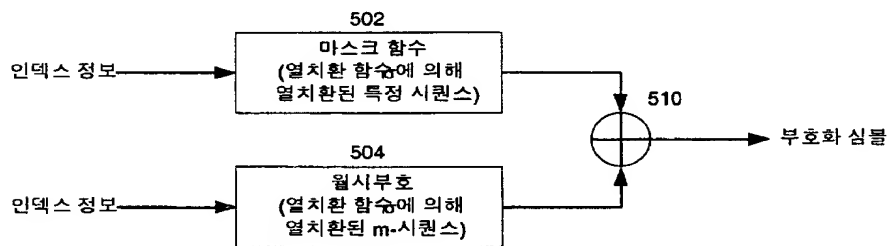
【도 3】



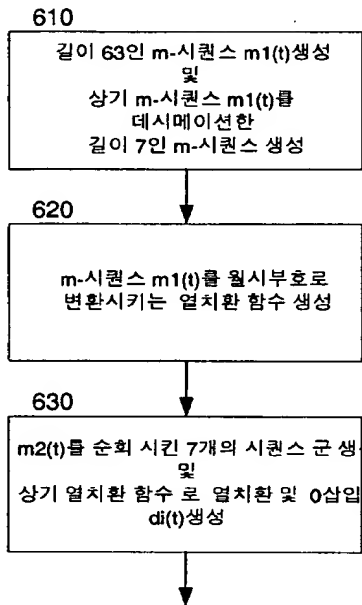
【도 4】



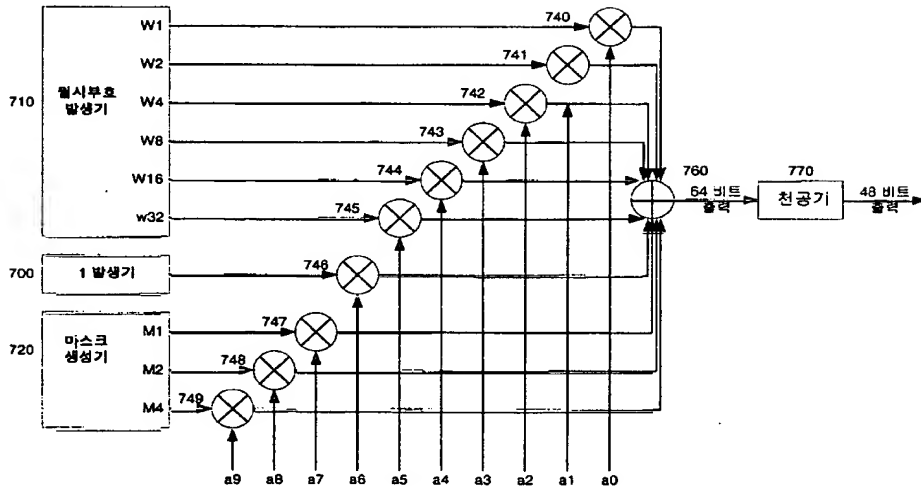
【도 5】



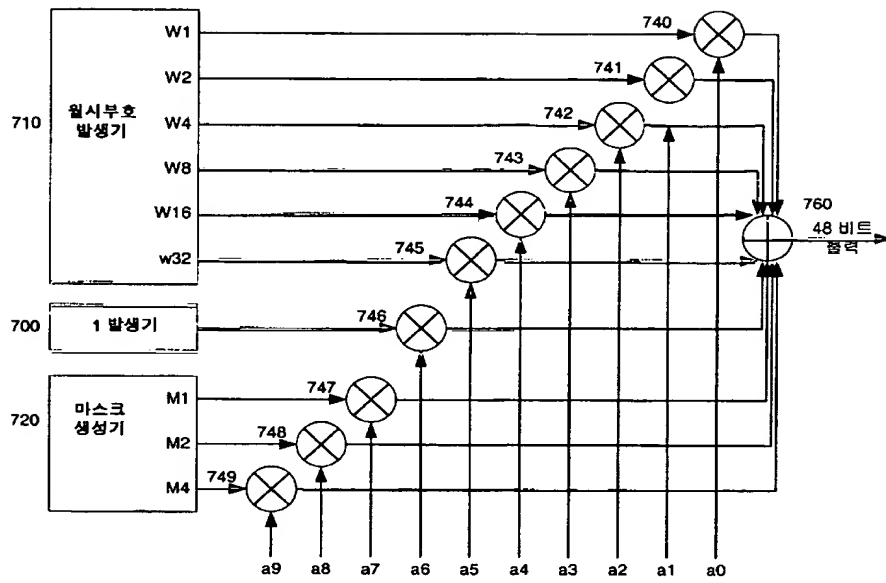
【도 6】



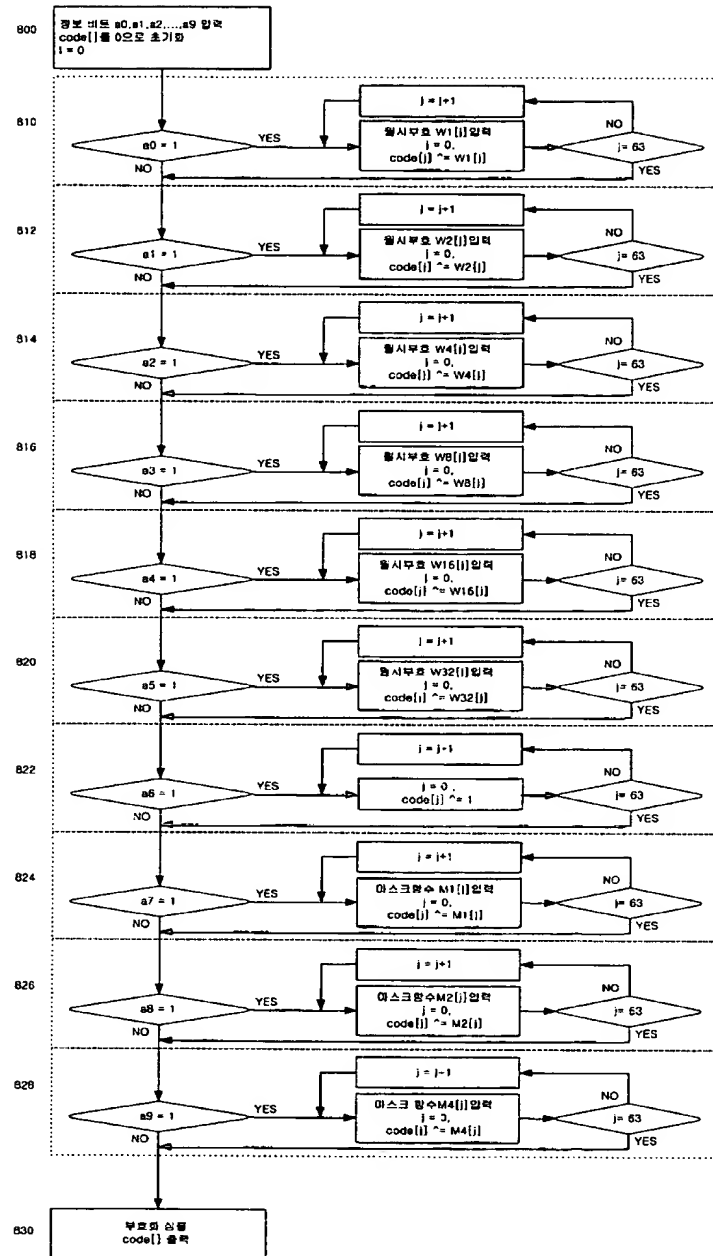
【도 7a】



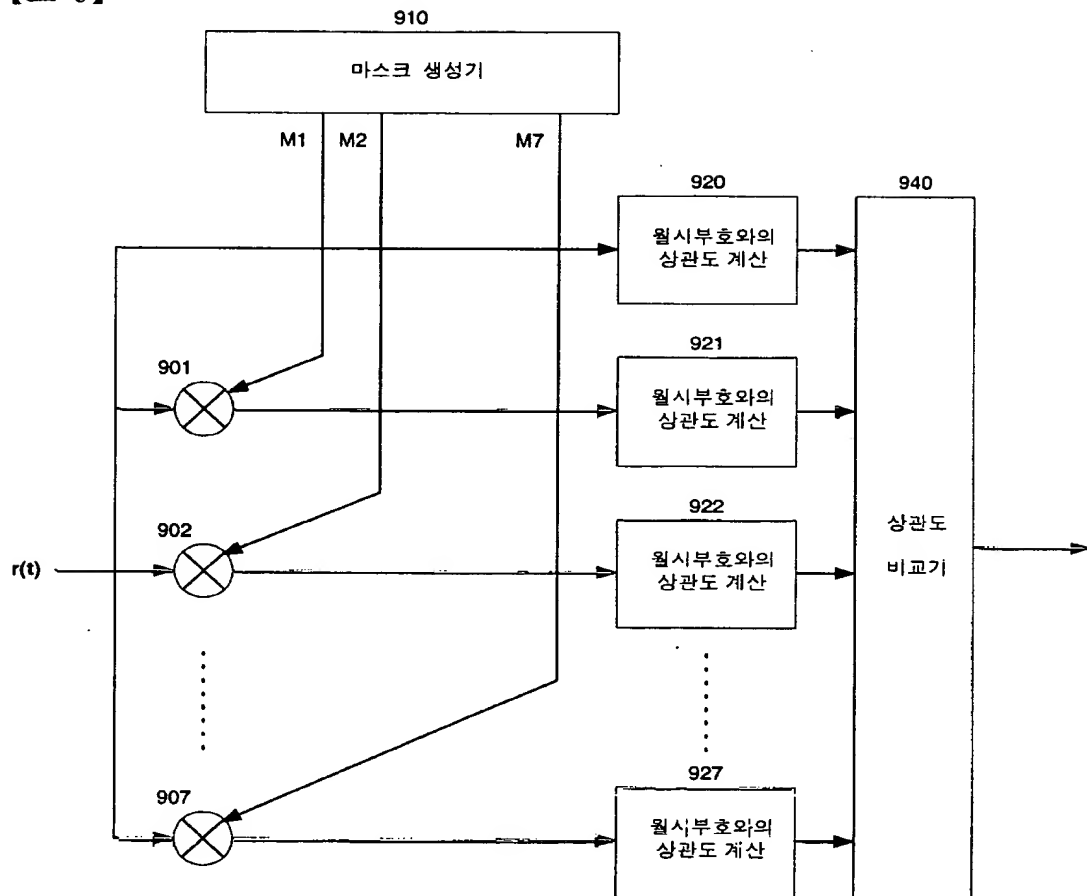
【도 7b】



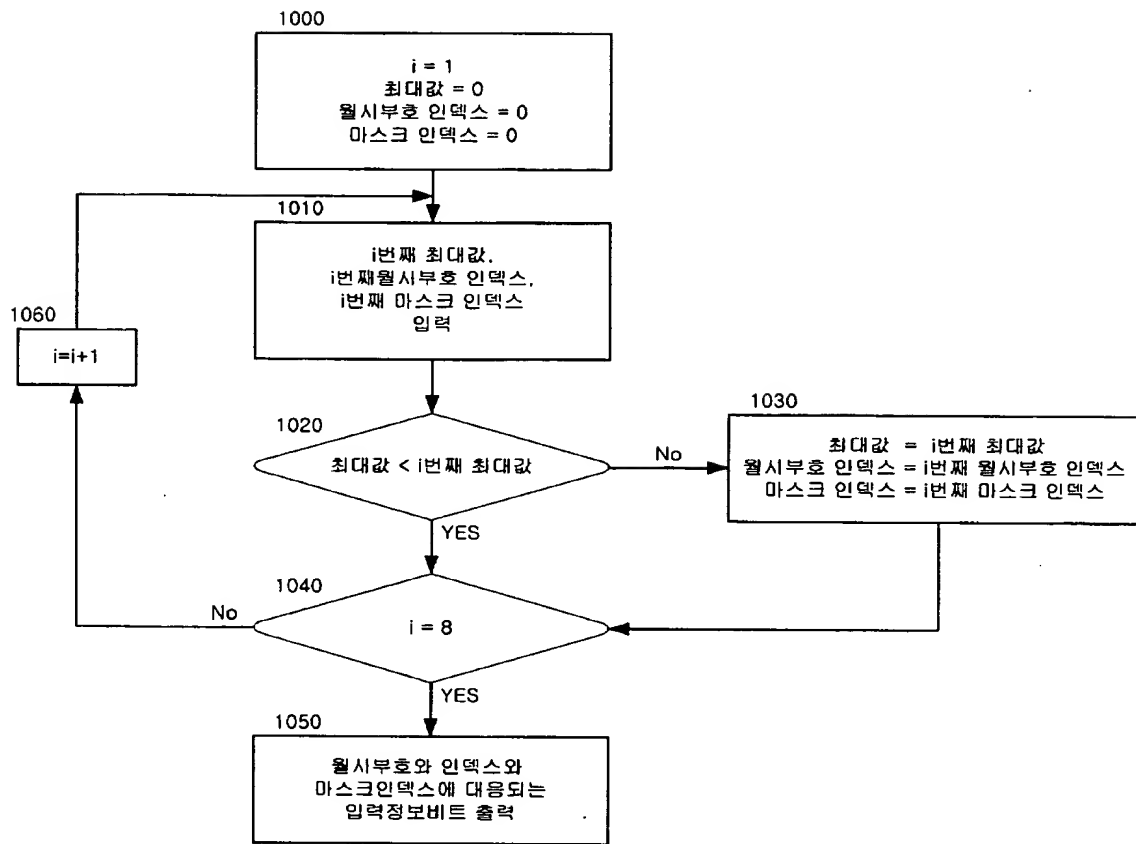
【도 8】



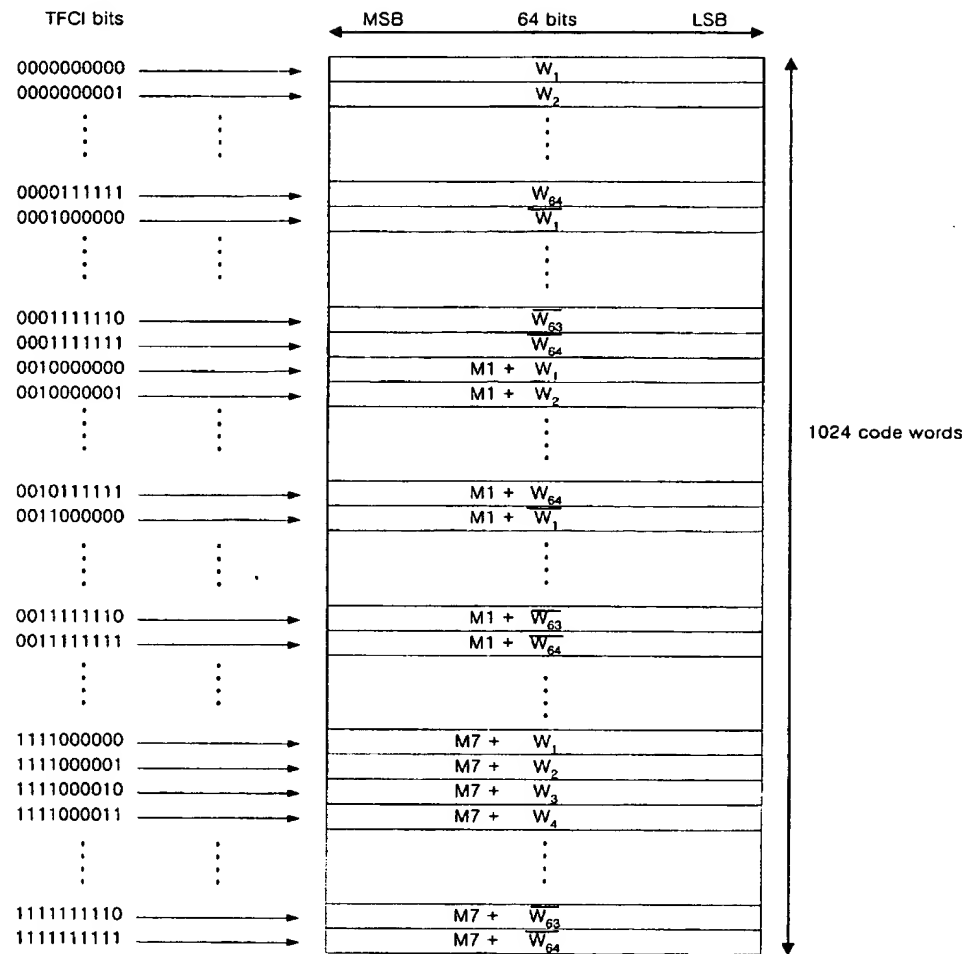
【도 9】



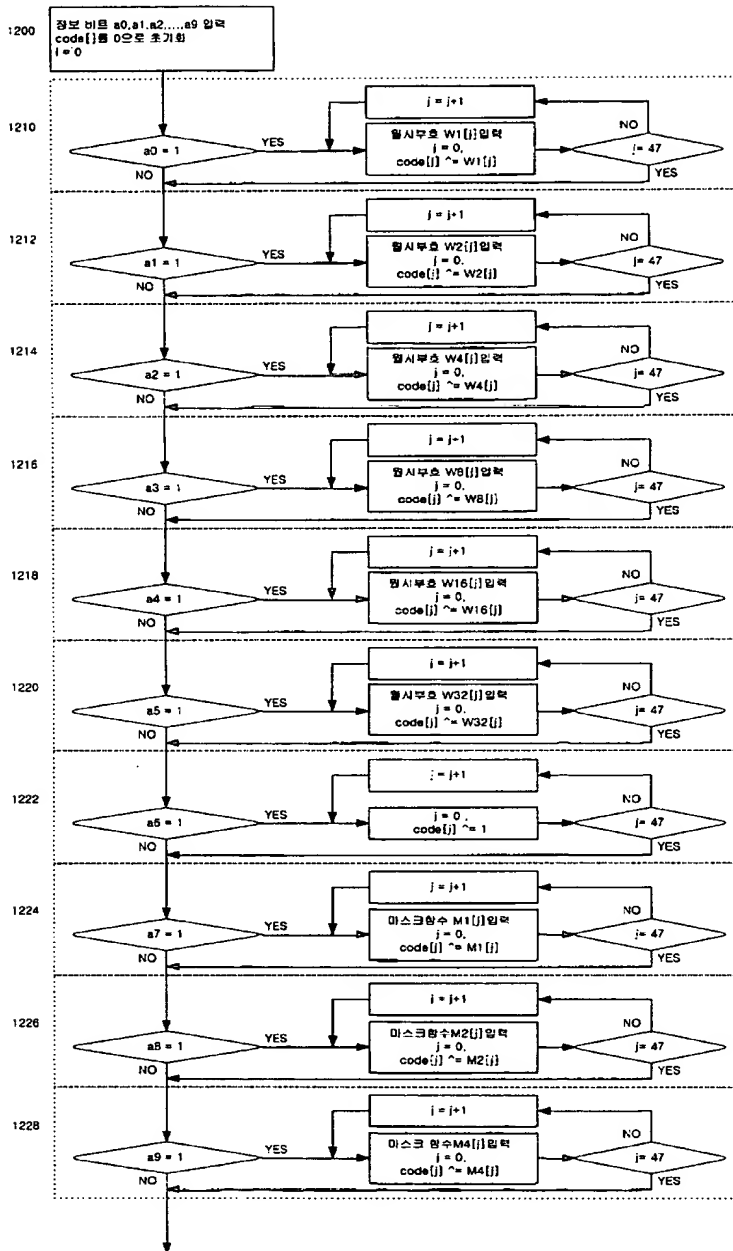
【도 10】



【도 11】



【도 12】



【서류명】	서지사항	보정서
【수신처】	특허청장	
【제출일자】	2000.09.15	
【제출인】		
【명칭】	삼성전자	주식회사
【출원인코드】	1-1998-104271-3	
【사건과의 관계】	출원인	
【대리인】		
【성명】	이건주	
【대리인코드】	9-1998-000339-8	
【포괄위임등록번호】	1999-006038-0	
【사건의 표시】		
【출원번호】	10-2000-0033107	
【출원일자】	2000.06.12	
【발명의 명칭】	무선통신시스템의	채널 부호화/복호화 장치 및 방법
【제출원인】		
【발송번호】	1-5-2000-0033668-15	
【발송일자】	2000.09.14	
【보정할 서류】	특허출원서	
【보정할 사항】		
【보정대상 항목】	수수료	
【보정방법】	납부	
【보정내용】	미납	수수료
【취지】	특허법시행규칙 제13조의 규정에 의하여 위와 같이 제출합니다. 대리인	
	이건주 (인)	
【수수료】		
【보정료】	11,000	원
【기타 수수료】	56,000	원
【합계】	67,000	원